Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 8881684, 12 pages
https://doi.org/10.1155/2021/8881684

Research Article

Path Planning for Smart Car Based on Dijkstra Algorithm and Dynamic Window Approach

Li-sang Liu,[1,2,3,4] **Jia-feng Lin**,[1] **Jin-xin Yao**,[1,3] **Dong-wei He**,[1,2] **Ji-shi Zheng**,[1,4] **Jing Huang**,[1,3] **and Peng Shi**[1,5]

[1]*School of Electronic, Electrical Engineering and Physics, Fujian University of Technology, Fuzhou 350118, China*
[2]*Fujian Key Laboratory of A.E.D, Fujian University of Technology, Fuzhou 350118, China*
[3]*National Demonstration Centre for Experimental Electronic Information and Electrical Technology Education, Fujian University of Technology, Fuzhou 350118, China*
[4]*School of Computer Science and Electronic Engineering, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, UK*
[5]*School of Electrical and Electronic Engineering, The University of Adelaide, Adelaide, SA 5005, Australia*

Correspondence should be addressed to Li-sang Liu; 199398190@qq.com

Received 23 July 2020; Revised 29 January 2021; Accepted 3 February 2021; Published 15 February 2021

Academic Editor: Pei-Wei Tsai

Path planning and obstacle avoidance are essential for autonomous driving cars. On the base of a self-constructed smart obstacle avoidance car, which used a LeTMC-520 depth camera and Jetson controller, this paper established a map of an unknown indoor environment based on depth information via SLAM technology. The Dijkstra algorithm is used as the global path planning algorithm and the dynamic window approach (DWA) as its local path planning algorithm, which are applied to the smart car, enabling it to successfully avoid obstacles from the planned initial position and reach the designated position. The tests on the smart car prove that the system can complete the functions of environment map establishment, path planning and navigation, and obstacle avoidance.

1. Introduction

In recent years, new industries such as cloud computing, big data, data center, virtual/augmented reality (VR/AR), 5G, artificial intelligence (AI), Internet of Things (IoT), and optical fiber sensing have emerged. These developments have changed our way of life and have simplified the completion of tasks that were difficult in the past [1]. Nowadays, Mobile Robot (MR) is widely used in various fields, such as military, industrial, agricultural, and many other applications [2]. They can replace humans to complete various difficult or dangerous operations in dangerous and special environments such as aviation, underwater, and tunnels. For example, MultiModal Mall Entertainment Robot (MuMMER) [3] can offer the public with an entertaining and engaging experience in an open-air plaza. Pepper [4] is a human-like robot, who can serve as a family companion robot for elder people. Atlas robot [5], developed by Boston Dynamics, can replace people to perform special tasks in both indoor and outdoor environments. Robots are required to work in more and more complex environments nowadays, such as shopping malls, city streets, hospitals, and train stations. Thus, to coexist harmoniously with people and other agents in these highly dynamic environments is one of the key problems. For autonomous driving robots, path planning and obstacle avoidance are the most basic functions. Traditional intelligent obstacle avoidance robots generally use binocular cameras or distance detectors to avoid obstacles. These robots have poor perception of the environment and high dependence on light and can only achieve obstacle avoidance in specific environments. In view of this, the new intelligent obstacle avoidance vehicle uses a depth camera, which combines machine image recognition and machine vision technology. It can fulfill lots of SLAM functions such as indoor mapping, automatic driving, and robotic arm grabbing on the robot operating system (ROS for short) [6]. The smart

obstacle avoidance vehicle can be used in safety inspection and evaluation fields, such as road quality inspection, bridge maintenance, industrial pipeline leakage, and location search. It can also replace people to complete logistics and transportation work in dangerous environments that are not suitable for humans to work. It can also perform military tasks such as surveillance, security patrol, pollutant collection, and hazardous material disposal in a more subtle manner.

Path planning is one of the key techniques for guiding the robot in dynamic environments. The goal of the path planning is to control the robot from the starting point to the target point, subject to the constraints such that the robot does not touch any obstacle throughout the process [7]. According to the target range of path planning, it can be divided into local path planning and global path planning [8]. Local path planning refers to the smart car overall unknown or partially unknown environmental data, and the environmental information is collected by the sensors in real time to determine the distribution of local obstacles, and the optimal path from the starting point to the target point is selected, so it is also called dynamic planning. Global path planning is also known as static planning, which refers to the smart car knowing all the geographic data and planning corresponding feasible paths based on this environmental information, also known as static planning.

At present, local path planning mainly includes artificial potential field method and dynamic window approach. The artificial potential field method is an abstract artificial force field, which assumes that a virtual gravitational field is formed between the target point and the car, and the gravitational force is inversely proportional to the distance between the car and the target point; a virtual repulsion field is formed between the obstacle and the car; the repulsive force is inversely proportional to the distance between the car and the obstacle; the smart car is regarded as a particle. The path optimization can be achieved by using the form of gravitation and repulsion functions. The planned path has fast convergence speed, smooth path, good real-time algorithm, and small amount of calculation. However, when facing with a complex dynamic environment, there will be a situation where the combined force is zero and the car stops moving during the movement of the smart car, so the design of the gravitational field is the key to the success of the algorithm. The dynamic window approach is to combine the sampled multiple sets of speed with the motion constraints of speed and acceleration, then evaluate the simulated multiple sets of trajectories through an evaluation function, and select the optimal speed corresponding to the trajectory as the drive of the car. Compared with the artificial potential field method, this approach has the characteristics of fast planning, safe and reliable, and good real-time performance [8–10].

A lot of researches have been done on global path planning, and many efficient algorithms have been proposed. These algorithms can be divided into three categories, Graph Search Algorithms, Random Sampling Algorithms, and Intelligent Bionic Algorithms. The Graph Search Algorithms mainly include the Dijkstra algorithm, A∗ algorithm, BFS algorithm (breadth-first search), and DFS algorithm (depth-first search). The Dijkstra algorithm and A∗ algorithms are widely implemented in the ROS (robot operating system) [11]. Although these methods are improved by reducing the number of searching grids through a heuristic estimation, the planning efficiency is inevitably low when the environment is complex, however. Random Sampling Algorithms include BIT (batch informed trees), RABIT (regionally accelerated batch informed trees), RRT (rapidly exploring random tree), and Risk-DTRRT (risk-based dual-tree rapidly exploring random tree). These algorithms are more efficient and widely used in dynamic high-dimensional environments. Intelligent Bionic Algorithms mainly include GA (genetic algorithm), ACO (ant colony algorithm), ABC (artificial bee colony algorithm), and PSO (particle swarm optimization algorithm). These methods simulate the evolutionary and biomimetic insects' behaviors. Wang et al. [12] proposed the optimization of the genetic algorithm-particle swarm optimization algorithm (OGA-PSO) to speed up the calculation and solve the shortcoming of locally optimal. Liu et al. [13] raised a combined algorithm with artificial potential field and geometric local optimization method, aiming to search a globally optimal path. Mac et al. [14] optimized constrained multiobjective PSO with an accelerated update methodology, which can shorten and smoothen the optimal global robot path.

This paper designs and implements the autonomous path planning of the smart car. The rest of this paper is organized as follows. In Section 2, we create a two-dimensional map of an unknown indoor environment by using the SLAM method based on depth information to simulate laser scanning data. The Dijkstra global path planning algorithm and DWA (dynamic window algorithm) local path planning algorithm are introduced in Section 3. Then, in Sections 4 and 5, we test our combined algorithm in a simulation environment and on practical self-built smart cars. The results show that the car can successfully avoid obstacles from the planned initial position and reach the designated position. Section 6 comes with our conclusion that both the algorithm and the system are proved to be effective and feasible.

2. Building a Map Based on Depth Information

SLAM technology is a very important part in the realization of intelligent obstacle avoidance vehicle technology. It is a technology for the intelligent vehicle to rely on various sensors to obtain information about the surrounding environment and establish a map of the environment, while using the created environment map to achieve reliable positioning. It mainly solves the problems such as the navigation of the vehicle in an unknown environment. This paper uses the SLAM method based on depth information to simulate laser scanning data to create a two-dimensional map of an unknown indoor environment [15].

First, obtain the color map and depth map of the indoor scene through the depth camera, and establish a geometric model of depth data, and then implement the function of simulating 2D laser scanning according to the geometric model of depth data, and create a 2D map of the surrounding environment.

Figure 1 is a schematic diagram of the geometric model of depth data. The pixel value of each pixel in the depth image of the environment collected by the depth camera has a corresponding depth value. This depth value is the vertical distance depth from the plane of the object to the plane of the depth camera, not the straight-line distance R between the camera and the object. Therefore, only through the corresponding geometric transformation can obtain the distance information of the object.

The method of geometric transformation is shown in

$$\begin{cases} D = \text{depth}[i][j], \\ R = \dfrac{D}{\sin\theta}, \end{cases} \tag{1}$$

where the depth value corresponding to the pixel in the i th row and jth column of the depth image is expressed as depth$[i][j]$ and θ represents the angle in the geometric model coordinates of the coordinate of the pixel in the depth image [9].

The principle of converting depth map to laser data is shown in Figure 2. In the schematic diagram of the depth map to laser data, for any depth image point $m(u, v, z)$ in the image (which is shown as red block in Figure 2), the steps to convert each pixel data $m(u, v, z)$ into laser data $M(x, y, z)$ are as follows:

(1) Convert the point of the depth image into a point under the depth camera coordinate system

(2) Calculate the angle $\angle$AOC between the line AO and CO

$$\theta = \arctan\left(\frac{x}{z}\right) \tag{2}$$

(3) $\angle$AOC needs to be mapped into the corresponding laser data. Assuming that the minimum range and maximum range $[\alpha, \beta]$ of the laser are known, the laser beam has N equal divisions, that is, the array laser$[N]$ can be used to represent the laser data. The index n of the point M projected into the array laser can be calculated using the following formula:

$$n = \frac{\theta - \alpha}{(\beta - \alpha)/N} = N \bullet \frac{\theta - \alpha}{\beta - \alpha} \tag{3}$$

The value of laser$[n]$ represents the distance r between the point C projected on the x-axis of the point M in the depth camera coordinate system and the optical center O of the camera, that is:

$$\text{laser}[n] = r = OC = \sqrt{z^2 + x^2}. \tag{4}$$

3. Obstacle Avoidance Algorithm

Obstacle avoidance algorithm is the key to navigation, while path planning is the basic condition of obstacle avoidance

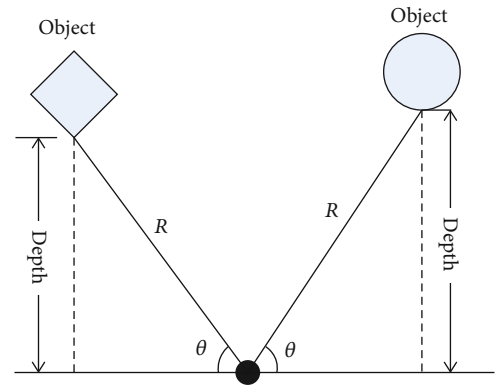


Figure 1: Schematic diagram of the geometric model of depth data.

algorithm. According to the target range of path planning, it can be divided into global path planning and local path planning. Global path planning refers to finding a good collision-free path among the known environment maps, and local path planning refers to planning of the intelligent car to realize its own positioning and dynamic obstacle avoidance through sensor information in a partially unknown or completely unknown environment map.

This paper uses the Dijkstra global path planning algorithm and DWA (dynamic window algorithm) local path planning algorithm. Based on the success of global path planning, that is, global path planning first plans a roughly feasible route and then maps the global path to the local map and then uses the DWA algorithm for local path planning according to the problem of unknown obstacles that the car may face. The local path planner selects or truncates the map mapped by the global path planner based on real-time information, then samples the velocity space, scores each generated trajectory to find the optimal trajectory, and then issues velocity instructions for path planning.

3.1. Global Path Planning Algorithm. Global path planning is to make the smart car plan a path smoothly from the starting point (where it is located) to the ending point (target point). In the process of global planning, first of all, the car is required to reach the target point accurately and cannot collide with obstacles in the environment, and secondly, it is necessary to choose the path that can reach the destination as quickly as possible. The global planner designed at this time uses the Dijkstra algorithm. After specifying a starting point and an ending point in the built environment map, the Dijkstra algorithm expands radially from the starting point to the outer layer until the target point and plans the shortest path between these two points.

When using the Dijkstra algorithm to plan the shortest path, it is usually necessary to specify the starting position of the car, then introduce two sets S and U. Set S is used to record the vertices of which the shortest path has not been found, and the distance from the vertex to the starting point [16].

(1) Initially, the starting point in the map is regarded as the set S, that is, the set S contains only the starting

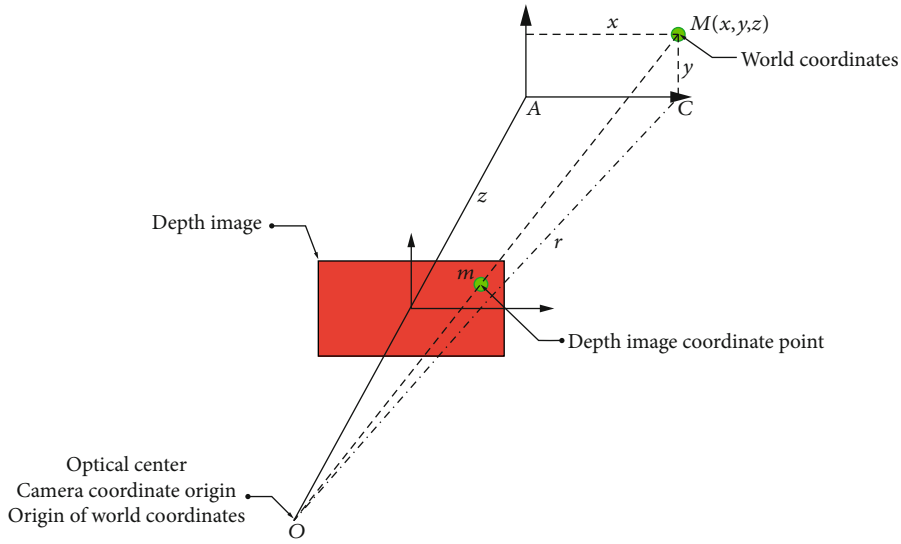


Figure 2: Schematic diagram of converting depth map to laser data.

point. The set U contains vertices other than the starting point

(2) According to the specified starting point, find the distance $d[i]$ from other points to the initial point. If the point is adjacent to the starting point, $d[i]$ is the edge weight (that is, the length between the two points); if the point is not adjacent to the starting point, $d[i]$ is ∞

(3) Select the smallest $d[i]$ from the set U (that is, the vertex with the shortest distance from the starting point), and add this vertex to the set S of vertices that have found the shortest path; at the same time, the vertex is removed from the set U of points that have never found the shortest path

(4) Update the distance from each vertex in the set U to the starting point

(5) Repeat steps (3) and (4) continuously until all vertices are searched; then, $d[i]$ corresponding to the target point is the shortest path length. The flowchart of the Dijkstra algorithm is shown in Figure 3

3.2. Local Path Planning Algorithm. Local path planning is to let the smart vehicle successfully complete "how to reach the destination" during the navigation process. Many uncertain factors may exist in practical car driving, for example, the local actual situation does not match the global map, temporary dynamic obstacles, and so on. It will be very dangerous if navigation was performed according to the global map. Thus, local path planning is necessary to avoid those dynamic or abrupt obstacles. Local path planning is mainly to avoid obstacles for local unknown obstacles and return to the calculated global path after avoiding obstacles. This paper uses the DWA algorithm to solve the local path planning problem.

The DWA is also called the dynamic window method, and its core idea is to transform the path planning problem into a constrained optimization problem on the velocity vector space. Multiple sets of velocities in the velocity space (v, ω), formed by the linear velocity v and the rotational velocity ω of the car, are sampled, and then, the trajectory is simulated. After obtaining multiple sets of trajectories, these trajectories are evaluated, and the optimal trajectory is selected as the actual trajectory of the car. However, there are certain constraints when sampling in velocity space. In addition, the obstacle cannot be regarded as a particle only, and the situation where the contour of the car may collide with the obstacle needs to be considered. Therefore, the dynamic window approach is added to the calculation of the expansion radius of the obstacle. The expansion radius is equal to the farthest length of the contour particle of the car and the edge of the obstacle outline. The flowchart of the DWA is shown in Figure 4. The steps of the dynamic window method are as follows:

(1) Sampling the data from the speed space to get the current state of the car

(2) For each sampling speed, calculate the movement trajectory of the car at this speed for a period

(3) Use the criteria of the evaluation function to score multiple routes and discard the infeasible routes

(4) The optimal path is selected as the actual trajectory of the car according to the scoring situation [17]

Note that the number of speed (v, ω) is infinite theoretically; thus, it is necessary to limit the speed sampling range [18]:

The car's velocity range in its own maximum and minimum speed

$$V_m = \{v \in [V_{\min}, V_{\max}], \omega \in [\omega_{\min}, \omega_{\max}]\}. \quad (5)$$

Due to the limited torque of the motor, the performance of the car will be affected by the motor drive module, so there

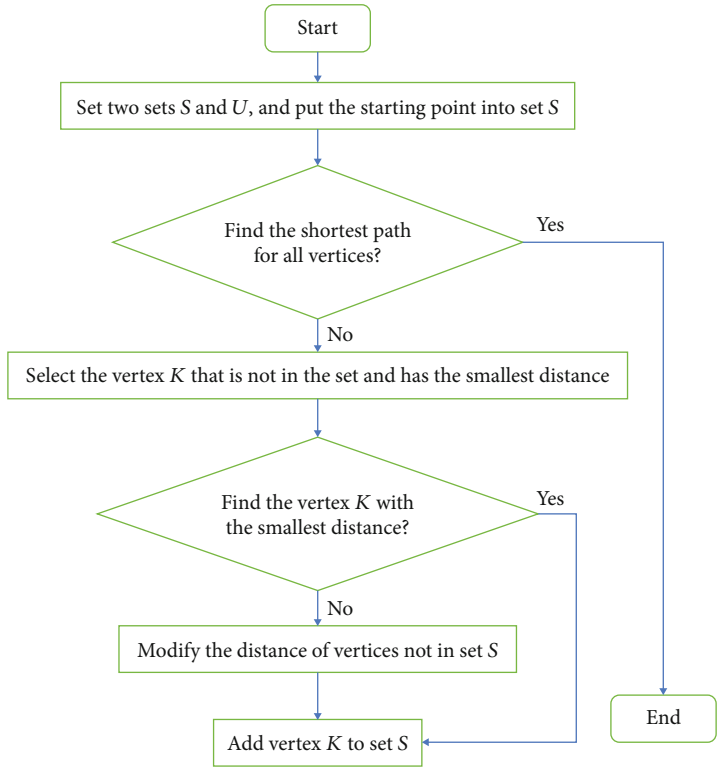


Figure 3: Flowchart of the Dijkstra algorithm.

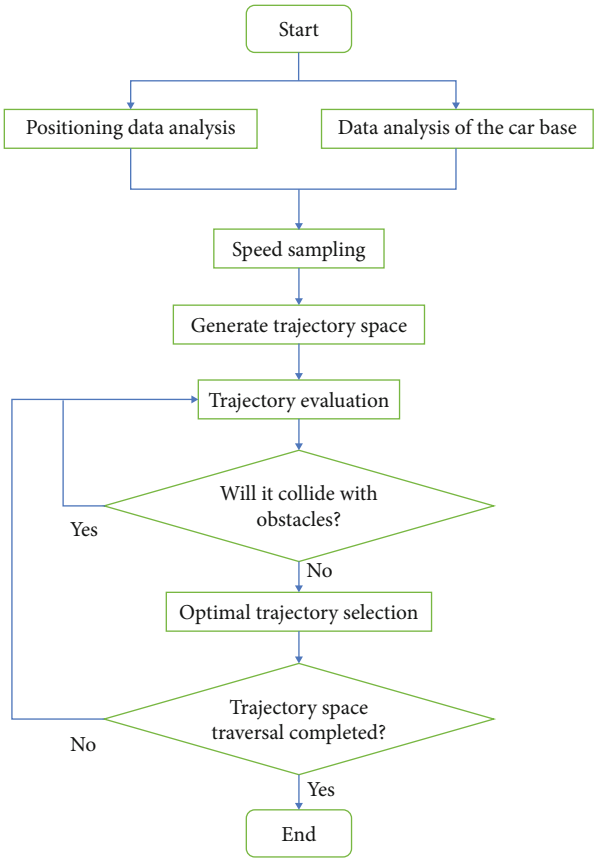


Figure 4: Flowchart of the DWA.

is a maximum acceleration and deceleration limit. Therefore, in the simulation cycle of the car traveling, the speed in the dynamic window is the speed that the car can actually reach:

$$V_d = \{(v, \omega) \mid v \in [V_c - V_b \Delta t, V_c + V_a \Delta t] \wedge \omega \in [w_c - w_b \Delta t, w_c + w_a \Delta t]\}. \quad (6)$$

Among them, V_c and ω_c are the current speed of the car, $V_a \Delta t$ and $w_a \Delta t$ correspond to the maximum acceleration, and $V_b \Delta t$ and $w_b \Delta t$ correspond to the maximum deceleration.

At the same time, in order to allow the car to stop before encountering an obstacle for safety consideration, the speed should have a range with the condition of maximum deceleration:

$$V_a = \left\{(v, \omega) \mid v \le \sqrt{2 \bullet \text{dist}(v, \omega) \bullet v_b} \wedge \omega \le \sqrt{2 \bullet \text{dist}(v, \omega) \bullet \omega_b}\right\}, \quad (7)$$

where $\text{dist}(v, \omega)$ represents the minimum distance between the car and the obstacle, and v_b and ω_b are the translation acceleration and rotation acceleration of the car, respectively. In the sampled speed group, there are several feasible trajectories. Therefore, each trajectory needs to be evaluated in the form of an evaluation function, so as to obtain the optimal value of the car at $k + 1$ time in the speed vector space.

The standard objective function of the DWA is

$$G(v, \omega) = \gamma \bullet \text{vel}(v, \omega) + \alpha \bullet \text{heading}(v, \omega) + \beta \bullet \text{distant}(v, \omega). \quad (8)$$

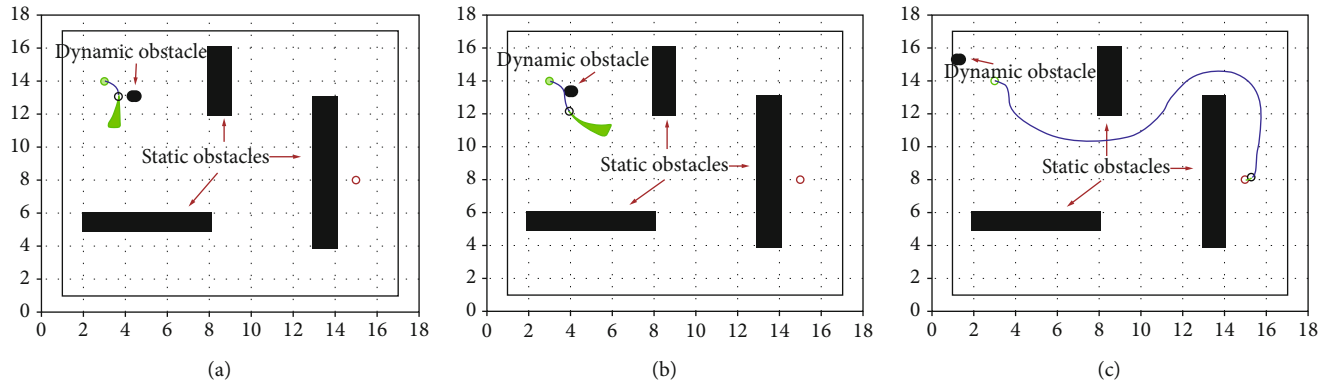


Figure 5: Simulation result of head-on situation: (a) the car meets the dynamic obstacle; (b) the car avoids the dynamic obstacle; (c) the car reaches the target position successfully.

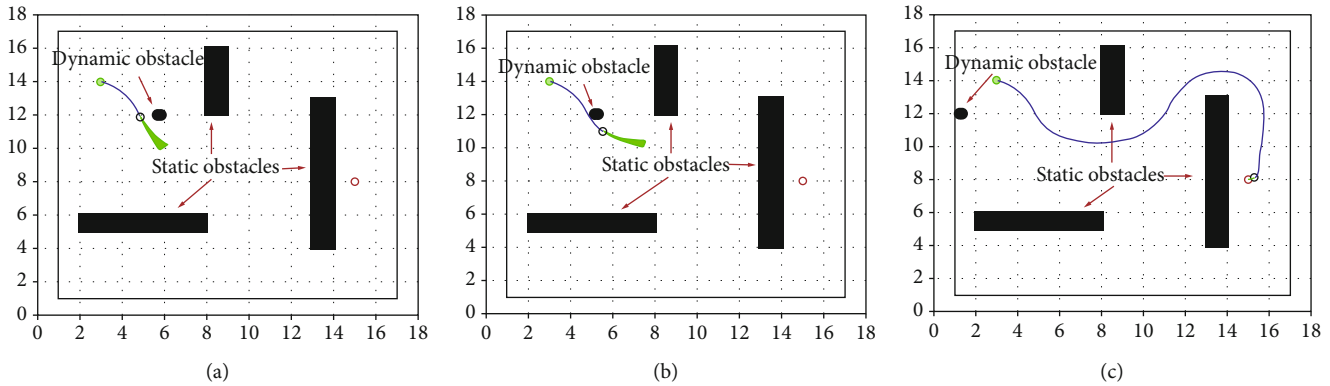


Figure 6: Simulation result of crossing situation: (a) the car meets the dynamic obstacle; (b) the car avoids the dynamic obstacle; (c) the car reaches the target position successfully.

The heading(v, ω) is a measure of progress towards the goal location. It is maximal if the car moves directly towards the target. When the angle between the car and the target point is 0 degree, the maximum value is

$$\text{heading}(v, \omega) = 1 - \frac{\theta}{\pi}. \tag{9}$$

Among them, θ represents the angle between the car and the target point.

The distant(v, ω) is the distance to the closest obstacle on the trajectory. L is the distance between the car's own position and the target point at that moment. The larger the value of the function, the farther the distance to the target point, and the less likely to be a collision; the smaller the value of the function, the closer to the obstacle, the greater the probability of a collision, which also means the higher is the car's desire to move around it.

$$\text{distant}(v, \omega) = \begin{cases} \frac{1}{L}, 0 \le l \le L, \\ 1, L \ll l. \end{cases} \tag{10}$$

The vel(v, ω) is the forward velocity of the car and supports fast movements.

$$\text{vel}(v, \omega) = \frac{v}{v_{\max}}, \tag{11}$$

where v is the average speed of the car and $v_{\max}$ is the maximum speed of the car in the speed space.

4. Navigation and Obstacle Avoidance Simulation

In order to verify the obstacle avoidance performance of the Dijkstra algorithm and DWA, we have established a 15×15 grid environment in MATLAB, as shown in Figures 5–7, and three groups of simulation experiments are carried out for navigation and avoidance of both static and dynamic obstacles. There are three different situations in each group of simulation, that is, the smart car meets dynamic obstacles in three ways, head-on, crossing, and overtaking.

The parameters in the simulation experiment are as follows: the maximum linear velocity and angular velocity of the smart car are set as 3.5 m/s and 40 rad/s; the maximum linear acceleration and angular acceleration are 0.35 m/s^2

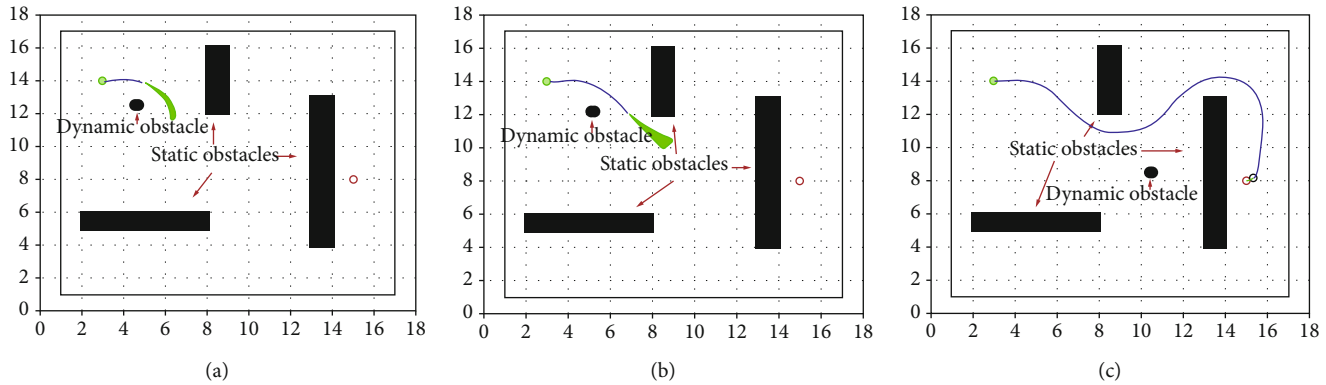


FIGURE 7: Simulation result of overtaking situation: (a) the car meets the dynamic obstacle; (b) the car avoids the dynamic obstacle; (c) the car reaches the target position successfully.

FIGURE 8: Smart obstacle avoidance car.

and 60 rad/s^2; the linear velocity resolution is 0.01 m/s, and the angular velocity resolution is 1 rad/s. The three weighted coefficients $\sigma\sigma$, $\beta\beta$, and $\gamma\sigma$ of the evaluation function are set as 0.05, 0.2, and 0.1. The starting position is indicated by a green dot, and its coordinates are (3,14); the target position is indicated by a red dot, and its coordinates are (15,8); the radius of dynamic obstacles is 0.3 m; the blue curves are the optimal paths obtained by the Dijkstra algorithm and DWA.

The simulation results of head-on situation are shown in Figure 5. The dynamic obstacle moves about 55° up to the left at a speed of 0.35 m/s from the initial position (6,12) and head-on to the car. When meeting the dynamic obstacle, as shown in Figures 5(a) and 5(b), the car will actively detour under the dynamic obstacle along the optimal trajectory evaluated by the DWA to avoid it. Then, the car continues to move towards the target position along the optimal trajectory. Finally, we can see from Figure 5(c) that the car reaches the target position safely and plans a smooth path.

The simulation results of crossing situation are shown in Figure 6; the dynamic obstacle moves horizontally to the left at a same speed from the initial position (7.5,12) and crosses the car. At this moment, the car will still actively detour under the dynamic obstacle along the optimal trajectory to avoid it as shown in Figures 6(a) and 6(b). Then, the car continues to move towards the target position along the optimal trajectory. Finally, we can also see from Figure 6(c) that the car reaches the target position safely and plans a smooth path, too.

The simulation results of overtaking situation are shown in Figure 7. The dynamic obstacle moves about 55° down to the right at a speed of 0.15 m/s from the initial position (4,13). When meeting the slow-moving dynamic obstacle, as shown in Figures 7(a) and 7(b), the car will actively detour to the left and surmount it to avoid it. Finally, we can still see from Figure 7(c) that the car reaches the target position safely and plans a smooth path.

We can find from the above three simulation results that the smart car can avoid dynamic obstacles well considering head-on, crossing, or overtaking situations, and the smooth final path the car planned still meets the global optimal performance.

5. Navigation and Obstacle Avoidance Tests

A self-built smart car was constructed as shown in Figure 8, mainly using Jetson Nano as the main controller, a motor with a reduction ratio of 30, a LeTMC-520 depth camera, and so on. Before the test starts, the angular velocity, linear velocity, and IMU of the car should be calibrated, and the PID parameters of the camera module and motor control module should be debugged. Various tests such as navigation and obstacle avoidance can be conducted only after they are confirmed to be correct and feasible. The working principle diagram of the car is shown in Figure 9.

5.1. Build Environment Map. When the depth camera and Rviz visual control platform are working, the grid map can be utilized and the keyboard can be used to control the car and collect environmental information. When using SLAM to model a new environment, it is necessary to manually control the rotation and forward speed of the intelligent car at medium level , so as to ensure the accuracy of created map.- After collecting all the environmental information, save the established map in time to facilitate future experiments.

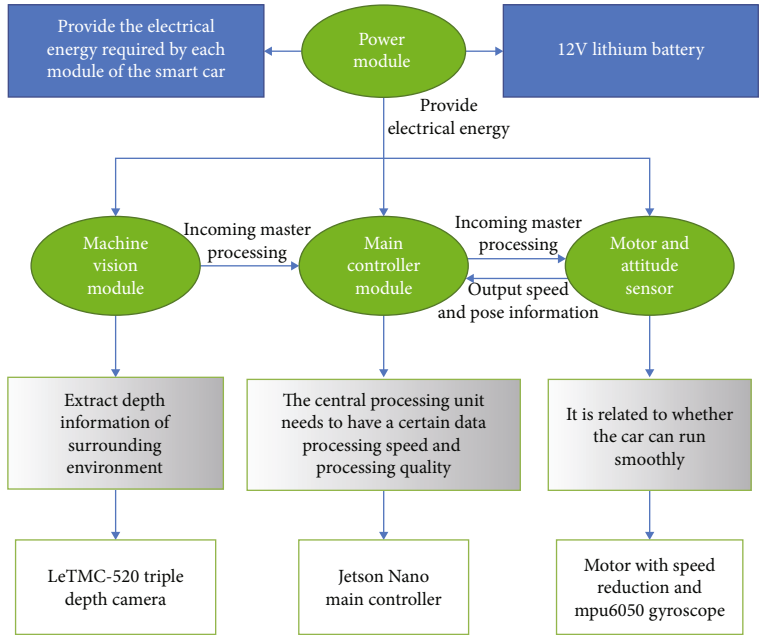


Figure 9: Principle of the smart car.

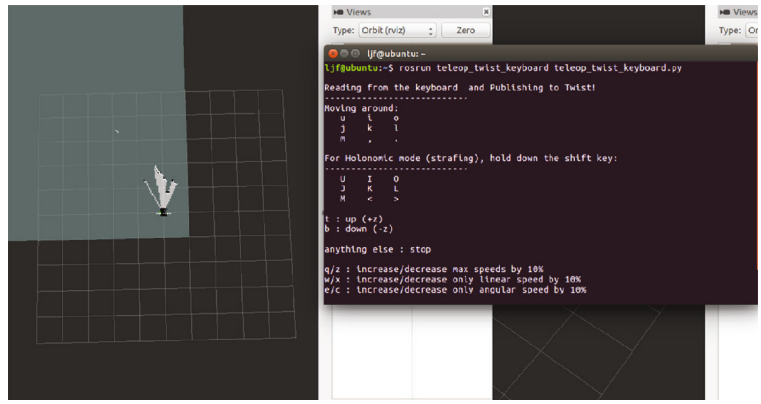


Figure 10: The image collected by the car at the original position.

The U, I, O, J, K, and L keys in the keyboard correspond to the left, straight, right, left turn, backward, and right turn of the control car, respectively.

Figure 10 is a map image collected when the car is just in its original position, showing the orientation of the car and the range of the environment currently being collected. Figure 11 is a built environment map of barrier-free information.

5.2. Obstacle Avoidance Tests. All the obstacle avoidance studies of this paper are based on the environment map. First, specify a starting position in the map, and then, use the global path planning and local path planning to make the car bypass the obstacle to reach the specified target position.

The test of the obstacle avoidance algorithm is divided into three situations, from simple to complex. To detect whether the obstacle avoidance algorithm is suitable for a variety of occasions, here are some representative detections as follows:

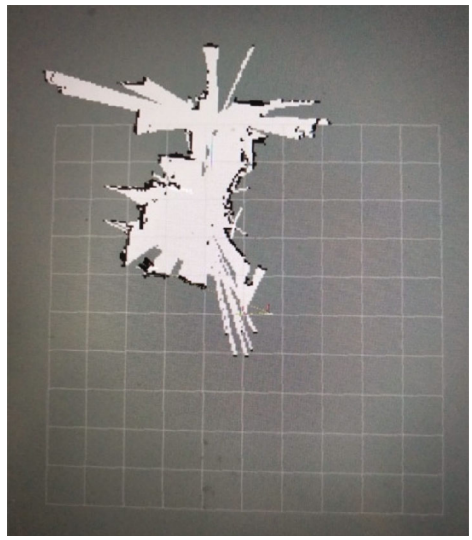
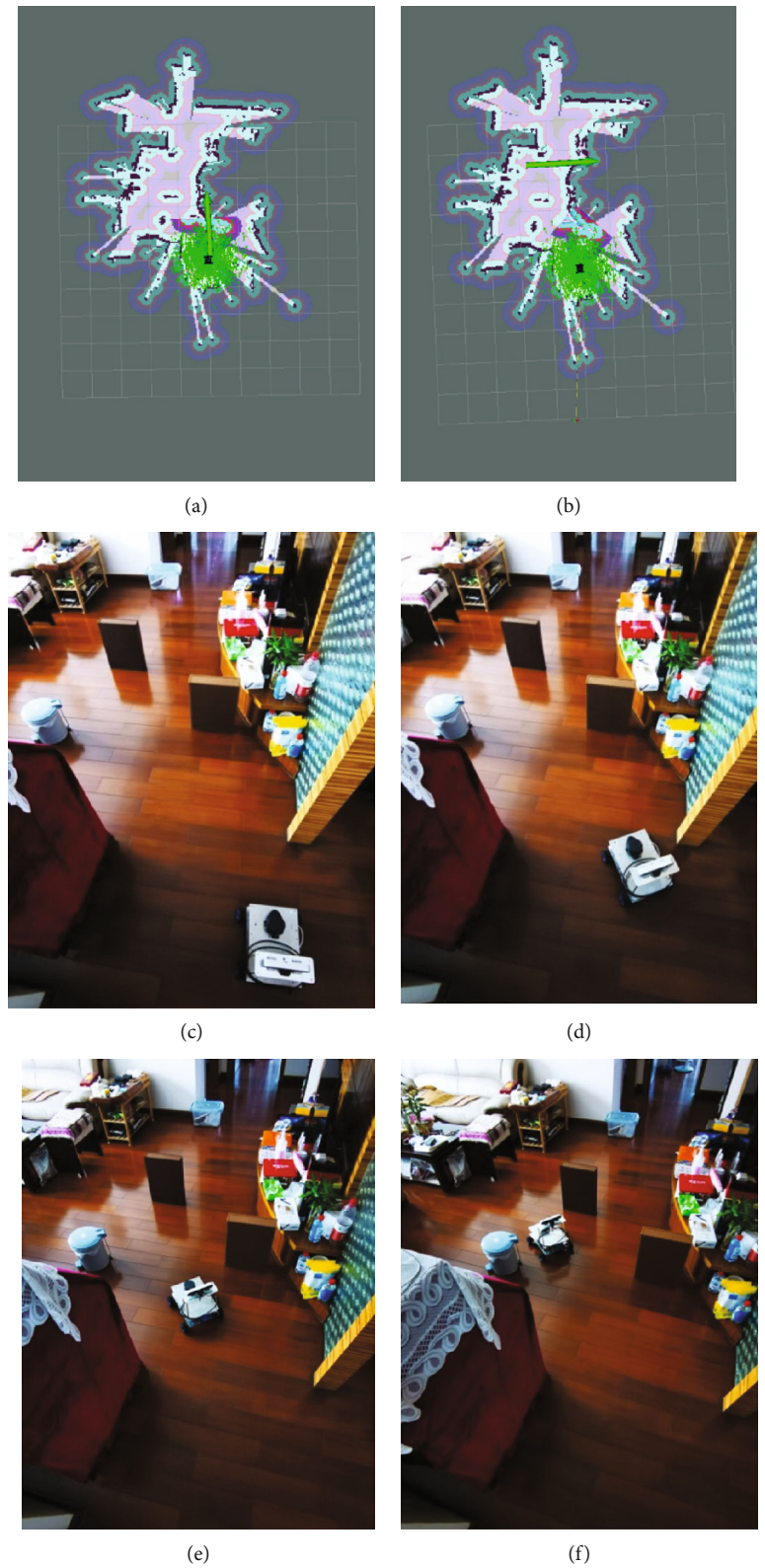
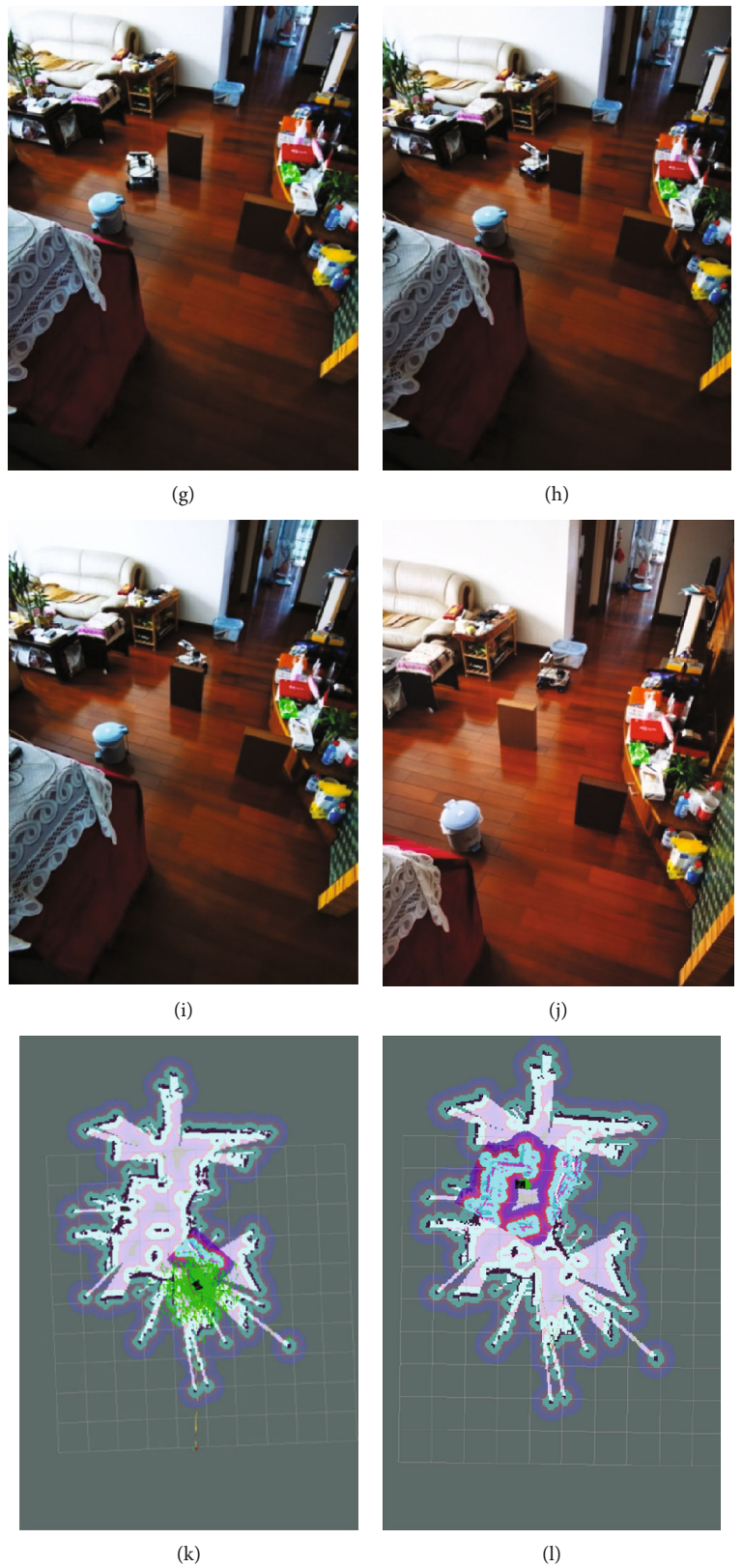


Figure 11: Environmental information of barrier-free objects.



(a) (b) (c) (d) (e) (f)

Figure 12: Continued.



(g) (h) (i) (j) (k) (l)

Figure 12: Obstacle avoidance test in a complex obstacle environment: (a) specifies the starting point; (b) specify the target point; (c) the starting position of the car; (d) bypass the first obstacle; (e) through box and trash can; (f) bypass trash can; (g) through the trash can and box; (h) adjust heading angle; (i) bypass box; (j) the car arrives at the designated location; (k) car path planning; (l) the console shows that the car has reached the destination (complete the task).

(1) In the simple obstacle environment, only one book is placed on the road as an obstacle. The test shows that the car can easily bypass obstacles from the starting position to the specified target position

(2) In a complicated environment, a book and two milk tanks are placed on the road, and which are placed in the road that are relatively close to each other enough for cars, to avoid obstacles to plan. The test also shows that the car can avoid obstacles and pass through the middle of these obstacles

(3) In a more complex environment, pianos, TV cabinets, and tea tables are used as borders on the road, and books, storage boxes, trash cans, and stools are used as obstacles. They are staggered to form an S-shaped roadblock. The specific obstacle avoidance test is shown in Figure 12. The test still shows that the car can also successfully avoid obstacles. It is worth mentioning that an appropriate inflation radius needs to be chosen carefully if there are different sizes of obstacles. In this experimental test, the value of the obstacle radius was finally set to 0.3 meters after debugging. If there are obstacles with special shapes and sizes, the obstacle radius value needs to be adjusted accordingly

The test process of the S-shaped roadblock in a complex environment is shown in Figure 12.

6. Conclusion

This paper designs and implements a smart obstacle avoidance car system, including hardware platform construction and software obstacle avoidance algorithm implementation. The main work can be summarized as follows: (1) A smart car was self-designed and self-built with a LeTMC-520 depth camera, a Jetson controller, four motors, and other modules. (2) Based on the ROS, SLAM technology was used to build a new unknown indoor environment map by converting depth map image to laser data; (3) the Dijkstra algorithm and DWA dynamic window method were used as the global path planning algorithm and local path planning algorithm, respectively. The validity and feasibility of the combined algorithm are approved by simulation results. The practical experiments were carried out and showed that the car can successfully avoid obstacles from the planned initial position and reach the designated position in a smooth planned path. The algorithm and the system are both proved to be effective and feasible.

Data Availability

Data are available on request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the Initial Scientific Research Fund of FJUT (GY-Z12079 and GY-Z160124), Fujian Provincial Education Department Youth Fund (JAT170367 and JAT170369), Natural Science Foundation of Fujian Province (2019J01773, 2018J01640, and 2017J01728), and China Scholarship Council (201709360002).

References

[1] Y. Zhang and J. Xin, "Survivable deployments of optical sensor networks against multiple failures and disasters: a survey," *Sensors*, vol. 19, no. 21, p. 4790, 2019.

[2] H. T. Nguyen and H. X. Le, "Path planning and obstacle avoidance approaches for mobile robot," *International Journal of Computer Science*, vol. 13, no. 4, 2016.

[3] M. E. Foster, R. Alami, O. Gestranius et al., "The MuMMER project: engaging human-robot interaction in real-world public spaces," in *Social Robotics. ICSR 2016. Lecture Notes in Computer Science, vol 9979*, A. Agah, J. J. Cabibihan, A. Howard, M. Salichs, and H. He, Eds., pp. 753–763, Springer, Cham, 2016.

[4] F. Tanaka, K. Isshiki, F. Takahashi, M. Uekusa, R. Sei, and K. Hayashi, "Pepper learns together with children: development of an educational application," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 270–275, Seoul, South Korea, November 2015.

[5] A. J. Bautista and S. O. Wane, "ATLAS robot: a teaching tool for autonomous agricultural mobile robotics," in *2018 International Conference on Control, Automation and Information Sciences (ICCAIS)*, pp. 264–269, Hangzhou, China, October 2018.

[6] T. Kaiqiao, *Design and Implementation in Intelligent Avoidance Obstacle Robot Based on ROS and Depth Camera Kinect*, Changchun University of Technology, 2018.

[7] K. Cai, C. Wang, J. Cheng, C. W. De Silva, and M. Q. Meng, "Mobile robot path planning in dynamic environments: a survey," *Instrumentation*, vol. 6, no. 2, pp. 92–102, 2019.

[8] H. Shuyun, T. Shoufeng, T. Ziyuan, S. Bin, and T. Minming, "A survey of path planning methods for autonomous mobile robots," *Software Guide*, vol. 17, no. 10, pp. 1–5, 2018.

[9] S. Xiaoru, R. Yiyue, G. Song, and C. Chaobo, "Survey on technology of mobile robot path planning," *Computer Measurement and Control*, vol. 27, no. 4, pp. 1–5+17, 2019.

[10] B. Encyclopedia, *Path planning [EB/OL]*2017-03-06, https://baike.baidu.com/item/pathplanning/8638339#3.

[11] L. Zhi and M. Xuesong, "Navigation and control system of mobile robot based on ROS," in *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 368–372, Chongqing, China, October 2018.

[12] X. Wang, Y. Shi, D. Ding, and X. Gu, "Double global optimum genetic algorithm–particle swarm optimization-based welding robot path planning," *Engineering Optimization*, vol. 48, no. 2, pp. 299–316, 2016.

[13] J. Liu, J. Yang, H. Liu, X. Tian, and M. Gao, "An improved ant colony algorithm for robot path planning," *Soft Computing*, vol. 21, no. 19, pp. 5829–5839, 2017.

[14] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "A hierarchical global path planning approach for mobile robots based

on multi-objective particle swarm optimization," *Applied Soft Computing*, vol. 59, pp. 68–76, 2017.

[15] W. Haozuo, *Research on SLAM Algorithm Based on Depth Perception of Indoor Mobile Robot*, Lanzhou University of Technology, 2016.

[16] S. Yong, *Research of Ship Path Planning Algorithm*, Wuhan University of Technology, 2018.

[17] C. Zhuo, S. Weihua, and S. Ning, "SLAM and path planning of mobile robot in ROS framework," *Medical and Health Equipment*, vol. 38, no. 2, pp. 109–113, 2017.

[18] W. Hanyuan, "Design and improvement of autonomous path planning algorithm for smart car," *Electronic Technology and Software Engineering*, vol. 4, pp. 84-85, 2018.